STA 445 HW2

Gabriel Cage-Sepeda

2024-02-22

Problem 1

Create a vector of three elements (2,4,6) and name that vector `vec_a`. Create a second vector, `vec_b`, that contains (8,10,12). Add these two vectors together and name the result `vec_c`.

```
vec_a <- c(2, 4, 6)
vec_b <- c(8, 10, 12)
vec_c <- vec_a + vec_b
```

Problem 2

Create a vector, named `vec_d`, that contains only two elements (14,20). Add this vector to `vec_a`. What is the result and what do you think R did (look up the recycling rule using Google)? What is the warning message that R gives you?

```
vec_d = c(14, 20)
vec_a + vec_d
```

```
## Warning in vec_a + vec_d: longer object length is not a multiple of shorter
## object length
```

```
## [1] 16 24 20
```

Error says that the longer vector is not a multiple of shorter vector. Output is length max(length_x, length_y) , and a warning will be thrown if the length of the longer vector is not an integer multiple of the length of the shorter vector.

Problem 3

Next add 5 to the vector vec_a. What is the result and what did R do? Why doesn't in give you a warning message similar to what you saw in the previous problem?

```
vec_a + 5
```

```
## [1]  7  9 11
```

R adds 5 to each element of vec_a. There is no warning because the length of vec_a, the longer object, is a multiple of the length of 5, the shorter object. That is, 3 is a multiple of 1.

Problem 4

Generate the vector of integers $\{1, 2, \ldots 5\}$ in two different ways.

a. First using the `seq()` function

```
seq(1,5)
```

```
## [1] 1 2 3 4 5
```

b. Using the `a:b` shortcut.

```
(1:5)
```

```
## [1] 1 2 3 4 5
```

Problem 5

Generate the vector of even numbers $\{2, 4, 6, \ldots, 20\}$

a. Using the seq() function

```
seq(2,20,2)
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

b. Using the a:b shortcut and some subsequent algebra.

```
(1:10)*2
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

Problem 6

Generate a vector of 21 elements that are evenly placed between 0 and 1 using the `seq()` command and name this vector `x`.

```
x <- seq(0, 1, 1/20)
```

Problem 7

Generate the vector $\{2, 4, 8, 2, 4, 8, 2, 4, 8\}$ using the `rep()` command to replicate the vector c(2,4,8).

```
rep(c(2, 4, 8), 3)
```

```
## [1] 2 4 8 2 4 8 2 4 8
```

Problem 8

Generate the vector $\{2, 2, 2, 2, 4, 4, 4, 4, 8, 8, 8, 8\}$ using the `rep()` command. You might need to check the help file for rep() to see all of the options that rep() will accept. In particular, look at the optional argument `each=`.

```
rep(c(2, 4, 8), 4, 12, 4)
```

```
##  [1] 2 2 2 2 4 4 4 4 8 8 8 8
```

Problem 9

In this problem, we will work with the matrix

$$\begin{bmatrix} 2 & 4 & 6 & 8 & 10 \\ 12 & 14 & 16 & 18 & 20 \\ 22 & 24 & 26 & 28 & 30 \end{bmatrix}$$

a. Create the matrix in two ways and save the resulting matrix as `M`.

b. Create the matrix using some combination of the `seq()` and `matrix()` commands.

```
M <- matrix(seq(2, 30, 2), nrow = 3, ncol = 5,
            byrow = TRUE)
```

ii. Create the same matrix by some combination of multiple `seq()` commands and either the `rbind()` or `cbind()` command.

```
M <- rbind(seq(2,10,2), seq(12, 20, 2), seq(22, 30, 2))
```

b. Extract the second row out of M.

```
M[2,]
```

```
## [1] 12 14 16 18 20
```

c. Extract the element in the third row and second column of M

```
M[3,2]
```

```
## [1] 24
```

Problem 10

The following code creates a `data.frame` and then has two different methods for removing the rows with `NA` values in the column `Grade`. Explain the difference between the two.

```
df <- data.frame(name= c('Alice','Bob','Charlie','Daniel'),
                      Grade = c(6,8,NA,9))

df[ -which(  is.na(df$Grade) ), ]

df[  which( !is.na(df$Grade) ), ]
```

The first way subtracts the row of any element that is na in the grade column. The second way includes every row that does not have an na element in the grade column.

Problem 11

Create and manipulate a list.

a. Create a list named my.test with elements + x = c(4,5,6,7,8,9,10) + y = c(34,35,41,40,45,47,51) + slope = 2.82 + p.value = 0.000131

```
my.test <- list(x = c(4,5,6,7,8,9,10),
                y = c(34,35,41,40,45,47,51),
                slope = 2.82, p.value = 0.000131)
```

b. Extract the second element in the list.

```
my.test$x[2]
```

```
## [1] 5
```

c. Extract the element named `p.value` from the list.

```
my.test$p.value
```

```
## [1] 0.000131
```